



[Subscribe \(Full Service\)](#) [Register \(Limited Service\)](#)

Search: ☒ The ACM Digital Library ☐ The ACM



[Feedback](#) [Report a problem](#) [Sign out](#)

Dynamic trace selection using performance monitoring hardware sampling

Full text [Pdf \(1.88 MB\)](#)

Source **ACM International Conference Proceeding Series; Vol. 37** [arch](#)
Proceedings of the international symposium on Code generation
feedback-directed and runtime optimization [table of contents](#)
San Francisco, California
SESSION: Profile-based optimizations [table of contents](#)
Pages: 79 - 90
Year of Publication: 2003
ISBN:0-7695-1913-X

Authors [Howard Chen](#) University of Minnesota
[Wei-Chung Hsu](#) University of Minnesota
[Jiwei Lu](#) University of Minnesota
[Pen-Chung Yew](#) University of Minnesota
[Dong-Yuan Chen](#) Microprocessor Research Labs, Intel

Sponsors : IEEE Computer Society TC-uARCH
[SIGMICRO](#): ACM Special Interest Group on Microarchitectural R
Processing

Publisher IEEE Computer Society Washington, DC, USA

Additional Information: [abstract](#) [references](#) [citings](#) [index terms](#) [collaborative](#)
[peer](#)

Tools and Actions: [Discussions](#) [Find similar Articles](#) [Review this Article](#)
[Save this Article to a Binder](#) Display Formats: [BibTex](#)

↑ ABSTRACT

Optimizing programs at run-time provides opportunities to apply aggressive c

programs based on information that was not available at compile time. At run time, we adapt to better exploit architectural features, optimize the use of dynamic instructions, and simplify code based on run-time constants. Our profiling system provides a framework for collecting information required for performing run-time optimization. We sample hardware registers available on an Itanium processor, and select a set of code blocks to monitor important performance-events. We gather distribution information about the events we wish to monitor, and test our traces by estimating the ability for dynamic instructions to execute run-time generated traces. Our results show that we are able to capture execution time across various SPEC2000 integer benchmarks using our profile and patching a relatively small number of frequently executed execution paths. Our profiling overhead increases execution time by only 2--4%.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full document. We have opted to expose the complete List rather than only correct and linked references.

- 1 Jennifer M. Anderson , Lance M. Berc , Jeffrey Dean , Sanjay Ghemawat , Henry Henzinger , Shun-Tak A. Leung , Richard L. Sites , Mark T. Vandevoorde , C. William E. Weihl, Continuous profiling: where have all the cycles gone?, *ACM Computer Systems (TOCS)*, v.15 n.4, p.357-390, Nov. 1997
- 2 Andrew Ayers , Stuart de Jong , John Peyton , Richard Schooler, Scalable optimization, *Proceedings of the ACM SIGPLAN 1998 conference on Program design and implementation*, p.301-312, June 17-19, 1998, Montreal, Quebec,
- 3 Vasanth Bala , Evelyn Duesterwald , Sanjeev Banerjia, Dynamo: a transparent optimization system, *Proceedings of the ACM SIGPLAN 2000 conference on language design and implementation*, p.1-12, June 18-21, 2000, Vancouver, B.C., Canada
- 4 Thomas Ball , James R. Larus, Efficient path profiling, *Proceedings of the ACM/IEEE international symposium on Microarchitecture*, p.46-57, December 1997, France
- 5 Pohua P. Chang , Scott A. Mahlke , Wen-mei W. Hwu, Using profile information for classic code optimizations, *Software—Practice & Experience*, v.21 n.12, p.13
- 6 Robert S. Cohn , David W. Goodwin , P. Geoffrey Lowney, Optimizing all

Windows NT with spike, Digital Technical Journal, v.9 n.4, p.3-20, April 1999

7 Thomas M. Conte , Burzin A. Patel , J. Stan Cox, Using branch handling hardware for profile-driven optimization, Proceedings of the 27th annual international symposium on Microarchitecture, p.12-21, November 30-December 02, 1994, San Jose, California

8 Daniel Conners, "Memory Profiling for Directing Data Speculative Optimization and Scheduling", Master Thesis, EE Department, University of Illinois, Urbana Champaign

9 Kemal Ebcioglu , Erik Altman , Michael Gschwind , Sumedh Sathaye, Dynamic Binary Translation and Optimization, IEEE Transactions on Computers, v.50 n.6, p.54-64, June 2001

10 Joseph A. Fisher , Stefan M. Freudenberger, Predicting conditional branch behavior from previous runs of a program, Proceedings of the fifth international conference on supercomputing support for programming languages and operating systems, p.85-95, October 1990, Boston, Massachusetts, United States

11 Susan L. Graham , Peter B. Kessler , Marshall K. McKusick, Gprof: A call graph based profiler, Proceedings of the 1982 SIGPLAN symposium on Compiler construction, June 23-25, 1982, Boston, Massachusetts, United States

12 Michael Gschwind , Erik R. Altman , Sumedh Sathaye , Paul Ledak , David L. Ditzler, Dynamic and Transparent Binary Translation, Computer, v.33 n.3, p.54-59, March 2000

13 Linley Gwennap, "Intel's Itanium and IA-64: Technology and Market Forecast", Intel Resources, Technical Library Report, 1999, http://www.mdronline.com/tech_library/ia64_techlib.htm

14 John L. Henning, SPEC CPU2000: Measuring CPU Performance in the Enterprise Environment, Computer, v.33 n.7, p.28-35, July 2000

15 Anne M. Holler, Optimization for a superscalar out-of-order machine, Proceedings of the annual ACM/IEEE international symposium on Microarchitecture, p.336-348, October 1996, Paris, France

16 Intel, "Intel IA-64" Architecture Software Developer's Manual, Volume 1: Architecture.

17 Intel, "Intel IA-64" Architecture Software Developer's Manual, Volume 2: Assembly Language Architecture.

- 18 Intel, "Intel IA-64" Architecture Software Developer's Manual, Volume 1: Programmer's Guide.
- 19 Thomas Kistler , Michael Franz, Continuous Program Optimization: Design and Implementation, IEEE Transactions on Computers, v.50 n.6, p.549-566, June 2001
- 20 T. Lafage and A. Seznec. Choosing representative slices of program execution for microarchitecture simulations: A preliminary application to the data stream. In Third IEEE Annual Workshop on Workload Characterization, pages 102--110, June 2000.
- 21 Scott A. Mahlke , David C. Lin , William Y. Chen , Richard E. Hank , Robert D. Diefenbach, Effective compiler support for predicated execution using the hyperblock, Proceedings of the 1999 annual international symposium on Microarchitecture, p.45-54, December 01-03, 1999, Portland, Oregon, United States
- 22 Matthew C. Merten , Andrew R. Trick , Erik M. Nystrom , Ronald D. Barham, A hardware mechanism for dynamic extraction and relay of program state, Proceedings of the 27th annual international symposium on Computer architecture, p.223-232, May 2000, Vancouver, British Columbia, Canada
- 23 Sanjay J. Patel , Steven S. Lumetta, rePLay: A Hardware Framework for Program Execution Optimization, IEEE Transactions on Computers, v.50 n.6, p.590-608, June 2001
- 24 Ashutosh S. Dhodapkar , James E. Smith, Managing multi-configuration program execution with dynamic working set analysis, Proceedings of the 29th annual international symposium on Computer architecture, p.233, May 25-29, 2002, Anchorage, Alaska
- 25 Eric Rotenberg , Steve Bennett , James E. Smith, Trace cache: a low latency, high bandwidth instruction fetching, Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture, p.24-35, December 02-04, 1996, Paris, France
- 26 Timothy Sherwood , Erez Perelman , Greg Hamerly , Brad Calder, Automatic characterization of large scale program behavior, Proceedings of the 10th international symposium on Architectural support for programming languages and operating systems, October 1997, San Jose, California
- 27 Timothy Sherwood , Erez Perelman , Brad Calder, Basic Block Distribution and Periodic Behavior and Simulation Points in Applications, Proceedings of the 1998 Conference on Parallel Architectures and Compilation Techniques, p.3-14, September 1998, New York, New York

- 28 S. Subramanya Sastry , Rastislav Bodík , James E. Smith, Rapid profiling sampling, Proceedings of the 28th annual international symposium on Computer Architecture, p.278-289, June 30-July 04, 2001, Göteborg, Sweden
- 29 Sun Microsystems, "MAJC-5200: A High Performance Microprocessor for Embedded Computing", white paper, <http://www.sun.com/microelectronics/MAJC/5200/>
- 30 Xiaolan Zhang , Zheng Wang , Nicholas Gloy , J. Bradley Chen , Michael J. Rabinovich, Support for automatic profiling and optimization, Proceedings of the sixteenth annual ACM symposium on Operating systems principles, p.15-26, October 05-08, 1997, Saint Malo, France
- 31 Yuan Chou , John Paul Shen, Instruction path coprocessors, Proceedings of the 27th annual ACM symposium on Computer architecture, p.270-281, June 2000, Vancouver, Canada

↑ CITINGS

Tipp Moseley , Alex Shye , Vijay Janapa Reddi , Matthew Iyer , Dan Fay , David L. D. Joshua L. Kihm , Alex Settle , Dirk Grunwald , Daniel A. Connors, Dynamic techniques for enabling continuous optimization, Proceedings of the 2nd conference on frontiers, May 04-06, 2005, Ischia, Italy

↑ INDEX TERMS

Primary Classification:

D. Software

↳ **D.3 PROGRAMMING LANGUAGES**

↳ **D.3.4 Processors**

↳ **Subjects: Optimization**

Additional Classification:

C. Computer Systems Organization

↳ **C.4 PERFORMANCE OF SYSTEMS**

↳ **Subjects: Measurement techniques; Performance attributes**

D. Software

↳ **D.3 PROGRAMMING LANGUAGES**

↪ **D.3.4 Processors**

↪ **Subjects:** Run-time environments

General Terms:

Algorithms, Performance

↑ **Collaborative Colleagues:**

<u>Dong-Yuan Chen:</u>	<u>Howard Chen</u> <u>Marina Chen</u> <u>Marina C. Chen</u> <u>Abhinav Das</u> <u>Jesse Fang</u> <u>Chen Fu</u> <u>Rao Fu</u> <u>Wei Chung Hsu</u> <u>Wei-Chung Hsu</u> <u>Yu Hu</u>	<u>Michel Jacquemin</u> <u>Roy Ju</u> <u>Jinpyo Kim</u> <u>Cheng-Yee Lin</u> <u>Lixia Liu</u> <u>Jiwei Lu</u> <u>Bobbie Othmer</u> <u>Chengyong Wu</u> <u>Jan-Jan Wu</u> <u>Youfeng Wu</u>	<u>Shuxin Yang</u> <u>Pen Chung Ye</u> <u>Pen-Chung Ye</u>
<u>Howard Chen:</u>	<u>Dong-Yuan Chen</u> <u>Abhinav Das</u> <u>Rao Fu</u> <u>Wei Chung Hsu</u> <u>Wei-Chung Hsu</u> <u>Jinpyo Kim</u> <u>Jiwei Lu</u> <u>Bobbie Othmer</u> <u>Pen Chung Yew</u> <u>Pen-Chung Yew</u>	<u>T. L. Yu</u>	
<u>Wei-Chung Hsu:</u>	<u>Santosh G. Abraham</u> <u>Sun Chan</u> <u>Dong-Yuan Chen</u> <u>Howard Chen</u> <u>Howard Howii Chen</u> <u>Tong Chen</u> <u>Xiaoru Dai</u> <u>Abhinav Das</u>	<u>Rao Fu</u> <u>James R. Goodman</u> <u>Edward H. Gornish</u> <u>Roy Dz-Ching Ju</u> <u>Jinpyo Kim</u> <u>Jin Lin</u> <u>Jiwei Lu</u> <u>Tin-Fook Ngai</u> <u>Khoa Nguyen</u> <u>Bobbie Othmer</u>	<u>Vatsa Santhanam</u> <u>James E. Smith</u> <u>Gurindar S. So</u> <u>Sriram Vajapeyam</u> <u>Pen-Chung Ye</u> <u>Antonia Zhai</u>




	<u>David A. Dunn</u>		
	<u>Charles N. Fisher</u>		
<u>Jiwei Lu:</u>	<u>Santosh G.</u>		
	<u>Abraham</u>		
	<u>Dong-Yuan Chen</u>		
	<u>Howard Chen</u>		
	<u>Abhinav Das</u>		
	<u>Rao Fu</u>		
	<u>Wei-Chung Hsu</u>		
	<u>Jinpyo Kim</u>		
	<u>Khoa Nguyen</u>		
	<u>Bobbie Othmer</u>		
	<u>Pen-Chung Yew</u>		
<u>Pen-Chung</u>	<u>Sarita V. Adve</u>	<u>Jesse Z. Fang</u>	<u>Gyungho Lee</u>
<u>Yew:</u>	<u>Christoffer Amlo</u>	<u>Zhixi Fang</u>	<u>Roland Lun Lo</u>
	<u>Doug Burger</u>	<u>Jeanne Ferrante</u>	<u>Sang-Jeong Lee</u>
	<u>Larry Carter</u>	<u>Rao Fu</u>	<u>J. H. Li</u>
	<u>Sun Chan</u>	<u>Catherine H.</u>	<u>Zhiyuan Li</u>
	<u>Siddhartha</u>	<u>Gebotys</u>	<u>J. Liang</u>
	<u>Chatterjee</u>	<u>Allan Gottlieb</u>	<u>David J. Lilja</u>
	<u>Ding-Kai Chen</u>	<u>Wei-Chung Hsu</u>	<u>Hock-Beng Li</u>
	<u>Dong-Yuan Chen</u>	<u>William Tsun-Yuk</u>	<u>Jin Lin</u>
	<u>Howard Chen</u>	<u>Hsu</u>	<u>Jiwei Lu</u>
	<u>T. Chen</u>	<u>B. Huang</u>	<u>Tin-Fook Nga</u>
	<u>Tong Chen</u>	<u>Chua-Huang Huang</u>	<u>Bobbie Othmer</u>
	<u>Sangyeun Cho</u>	<u>Jian Huang</u>	<u>David K. Poul</u>
	<u>Lynn Choi</u>	<u>Kai Hwang</u>	<u>Jan Prins</u>
	<u>Alok N.</u>	<u>Zhenzhen Jiang</u>	<u>Alasdair</u>
	<u>Choudhary</u>	<u>Roy Dz-Ching Ju</u>	<u>Rawsthorne</u>
	<u>Xiaoru Dai</u>	<u>David Kaeli</u>	<u>P. Sadayappan</u>
	<u>Abhinav Das</u>	<u>Mahmut T.</u>	<u>Sartaj Sahni</u>
	<u>Timothy A. Davis</u>	<u>Kandemir</u>	<u>Stephen J.</u>
	<u>Timothy Alden</u>	<u>Jinpyo Kim</u>	<u>Schwinn</u>
	<u>Davis</u>	<u>Pavlos Konas</u>	<u>David C. Sehr</u>
	<u>Y. H. Ding</u>	<u>Jinseok Kong</u>	<u>Michael D. Sn</u>
	<u>Rudolf Eigenmann</u>	<u>Duncan H. Lawrie</u>	

↑ **Peer to Peer - Readers of this Article have also read:**

- Data structures for quadtree approximation and compression
Communications of the ACM 28, 9
Hanan Samet
- A hierarchical single-key-lock access control using the Chinese remainder
Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing
Kim S. Lee , Huizhu Lu , D. D. Fisher
- The GemStone object database management system
Communications of the ACM 34, 10
Paul Butterworth , Allen Otis , Jacob Stein
- Putting innovation to work: adoption strategies for multimedia communication
Communications of the ACM 34, 12
Ellen Francik , Susan Ehrlich Rudman , Donna Cooper , Stephen Levine
- An intelligent component database for behavioral synthesis
Proceedings of the 27th ACM/IEEE conference on Design automation
Gwo-Dong Chen , Daniel D. Gajski

The ACM Portal is published by the Association for Computing Machinery.
ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)



[Subscribe](#) (Full Service) [Register](#) (Limited Ser

Search: ☐ The ACM Digital Library ☐ The



[Feedback](#) [Report a problem](#) [S](#)

Dynamic hot data stream prefetching for general-purpose programs

Full text [Pdf](#) (211 KB)

Source **Conference on Programming Language Design and Implement**
Proceedings of the ACM SIGPLAN 2002 Conference on Progr
design and implementation [table of contents](#)

Berlin, Germany

SESSION: Dynamic Prefetching & Cache Optimizations [table of c](#)

Pages: 199 - 209

Year of Publication: 2002

ISSN:0362-1340

[Also published in ...](#)

Authors [Trishul M. Chilimbi](#) Microsoft Research, Redmond, WA
[Martin Hirzel](#) University of Colorado, Boulder, CO

Sponsor [SIGPLAN](#): ACM Special Interest Group on Programming Language

Publisher ACM Press New York, NY, USA

Additional Information: [abstract](#) [references](#) [citings](#) [index terms](#) [collaborative](#)
[peer](#)

Tools and Actions: [Discussions](#) [Find similar Articles](#) [Review this Article](#)
[Save this Article to a Binder](#) Display Formats: [BibTex](#)

DOI Bookmark: Use this link to bookmark this Article: <http://doi.acm.org/10.1>
[What is a DOI?](#)

↑ ABSTRACT

Prefetching data ahead of use has the potential to tolerate the growing process performance gap by overlapping long latency memory accesses with useful co
sophisticated prefetching techniques have been automated for limited domain

codes that access dense arrays in loop nests, a similar level of success has eluded programs, especially pointer-chasing codes written in languages such as C and C++. This problem is addressed by describing, implementing and evaluating a dynamic prefetching technique that runs on stock hardware, is completely automatic, and works for general-purpose programs, including pointer-chasing codes written in weakly-typed languages. It operates in three phases. First, the profiling phase gathers a temporal data reference profile from a running program with low-overhead. Next, the profiling is turned off and an analysis algorithm extracts hot data streams, which are data reference sequences that follow the same order, from the temporal profile. Then, the system dynamically injects appropriate program points to detect and prefetch these hot data streams. Finally, the program enters the hibernation phase where no profiling or analysis is performed, and it continues to execute with the added prefetch instructions. At the end of the hibernation phase, the program is de-optimized to remove the inserted checks and prefetch instructions and returns to the profiling phase. For long-running programs, this profile, analysis, hibernation, and cycle will repeat multiple times. Our initial results from applying this technique are promising, indicating overall execution time improvements of 5.19% for several performance-limited SPECint2000 benchmarks running their largest (ref) inputs.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full paper. The author has opted to expose the complete List rather than only correct and linked references.

- 1 Murali Annavaram , Jignesh M. Patel , Edward S. Davidson, Data prefetching using a graph precomputation, Proceedings of the 28th annual international symposium on computer architecture, p.52-61, June 30-July 04, 2001, Göteborg, Sweden
- 2 Matthew Arnold , Stephen Fink , David Grove , Michael Hind , Peter F. Sweeney, Static optimization in the Jalapeño JVM, Proceedings of the 15th ACM SIGPLAN conference on programming languages, systems, languages, and applications, p.47-65, October 1998, Minneapolis, Minnesota, United States
- 3 Matthew Arnold , Barbara G. Ryder, A framework for reducing the cost of dynamic optimization, Proceedings of the ACM SIGPLAN 2001 conference on Programming language implementation, p.168-179, June 2001, Snowbird, Utah, United States
- 4 Vasanth Bala , Evelyn Duesterwald , Sanjeev Banerjia, Dynamo: a transparent dynamic optimization system, Proceedings of the ACM SIGPLAN 2000 conference on

language design and implementation, p.1-12, June 18-21, 2000, Vancouver, E Canada

5 Brendon Cahoon , Kathryn S. McKinley, Data Flow Analysis for Software Data Structures in Java, Proceedings of the 2001 International Conference on Architectures and Compilation Techniques, p.280-291, September 08-12, 2001

6 M. Charney, and A. Reeves. "Generalized correlation based hardware prefetching", EE-CEG-95-1, Cornell University, 1995

7 Tien-Fu Chen , Jean-Loup Baer, Reducing memory latency via non-blocking caches, Proceedings of the fifth international conference on Architectural support for programming languages and operating systems, p.51-61, October 12-15, 1992, Boston, Massachusetts

8 Trishul M. Chilimbi, Efficient representations and abstractions for quantifying data reference locality, Proceedings of the ACM SIGPLAN 2001 conference on language design and implementation, p.191-202, June 2001, Snowbird, Utah,

9 Trishul M. Chilimbi , James R. Larus, Using generational garbage collection to reduce cache-conscious data placement, Proceedings of the 1st international symposium on garbage collection, p.37-48, October 17-19, 1998, Vancouver, British Columbia, Canada

10 Trishul M. Chilimbi, On the Stability of Temporal Data Reference Profiles, Proceedings of the 2001 International Conference on Parallel Architectures and Compilation Techniques, p.1-12, September 08-12, 2001

11 Michał Cierniak , Guei-Yuan Lueh , James M. Stichnoth, Practicing JUD for dynamic optimizations, Proceedings of the ACM SIGPLAN 2000 conference on language design and implementation, p.13-26, June 18-21, 2000, Vancouver, Canada

12 Robert Cooksey , Dennis Colarelli , Dirk Grunwald, Content-Based Prefetching: Results, Revised Papers from the Second International Workshop on Intelligent Prefetching, p.33-55, November 12, 2000

13 D. Deaver, R. Gorton, and N. Rubin, "Wiggins/Redstone: An online program for Hot Chips, 1999

14 Timothy L. Harris, Dynamic adaptive pre-tenuring, Proceedings of the symposium on dynamic languages, p.1-12, June 18-21, 2000, Vancouver, E Canada

symposium on Memory management, p.127-136, October 15-16, 2000, Minneapolis, United States

15 M. Hirzel and T. Chilimbi. "Bursty Tracing: A Framework for Low-Overhead Profiling", In Workshop on Feedback-Directed and Dynamic Optimizations

16 Doug Joseph , Dirk Grunwald, Prefetching using Markov predictors, Proceedings of the annual international symposium on Computer architecture, p.252-263, June 01-03, 1998, Colorado, United States

17 Norman P. Jouppi, Improving direct-mapped cache performance by the addition of a fully-associative cache and prefetch buffers, Proceedings of the 17th annual international symposium on Computer Architecture, p.364-373, May 28-31, 1990, Seattle, Washington, United States

18 M. Karlsson, F. Dahlgren, and P. Stenstrom. "A Prefetching Technique for Data Access to Linked Data Structures, In High Performance Computer Architectures (HPCA)

19 Thomas Kistler , Michael Franz, Automated data-member layout of heap memory-hierarchy performance, ACM Transactions on Programming Language and Systems (TOPLAS), v.22 n.3, p.490-505, May 2000

20 Alexander C. Klaiber , Henry M. Levy, An architecture for software-controlled prefetching, Proceedings of the 18th annual international symposium on Computer Architecture, p.43-53, May 27-30, 1991, Toronto, Ontario, Canada

21 James R. Larus, Whole program paths, Proceedings of the ACM SIGPLAN conference on Programming language design and implementation, p.259-269, May 01-04, 1990, Atlanta, Georgia, United States

22 Chi-Keung Luk , Todd C. Mowry, Compiler-based prefetching for recursive data structures, Proceedings of the seventh international conference on Architectural support for programming languages and operating systems, p.222-233, October 01-04, 1996, Cambridge, Massachusetts, United States

23 Craig G. Nevill-Manning , Ian H. Witten, Linear-Time, Incremental Hierarchical Data Compression, Proceedings of the Conference on Data Compression, p.3, March 1996, Boston, Massachusetts, United States

24 Todd C. Mowry , Monica S. Lam , Anoop Gupta, Design and evaluation of a hardware algorithm for prefetching, Proceedings of the fifth international conference on

support for programming languages and operating systems, p.62-73, October
Massachusetts, United States

25 M. Paleczny, C. Vick, and C. Click. "The Java HotSpot server compiler."
Virtual Machine Research and Technology Symposium (JVM), 2001

26 Amir Roth , Andreas Moshovos , Gurindar S. Sohi, Dependence based pr
data structures, Proceedings of the eighth international conference on Archite
programming languages and operating systems, p.115-126, October 02-07, 19
California, United States

27 Amir Roth , Gurindar S. Sohi, Effective jump-pointer prefetching for link
Proceedings of the 26th annual international symposium on Computer archite
May 01-04, 1999, Atlanta, Georgia, United States

28 Shai Rubin , Rastislav Bodík , Trishul Chilimbi, An efficient profile-anal
data-layout optimizations, Proceedings of the 29th ACM SIGPLAN-SIGACT
Principles of programming languages, p.140-153, January 16-18, 2002, Portla

29 Rafael H. Saavedra , Daeyeon Park, Improving the Effectiveness of Softw
Adaptive Execution, Proceedings of the 1996 Conference on Parallel Architec
Compilation Techniques (PACT '96), p.68, October 20-23, 1996

30 Timothy Sherwood , Brad Calder, Automated design of finite state machi
customized processors, Proceedings of the 28th annual international symposi
architecture, p.86-97, June 30-July 04, 2001, Göteborg, Sweden

31 Amitabh Srivastava , Alan Eustace, ATOM: a system for building custom
analysis tools, Proceedings of the ACM SIGPLAN 1994 conference on Progr
design and implementation, p.196-205, June 20-24, 1994, Orlando, Florida, U

32 A. Srivastava, A. Edwards, and H. Vo. "Vulcan: Binary trans¿formation i
environment.", In Microsoft Re¿search Tech Report, MSR-TR-2001-50, 2001

33 Artour Stoutchinin , José N. Amaral , Guang R. Gao , James C. Dehnert ,
Douillet, Speculative Prefetching of Induction Pointers, Proceedings of the 10
Conference on Compiler Construction, p.289-303, April 02-06, 2001

34 D. Ung, and C. Cifuentes. "Opimising hot paths in a dynamic binary trans
on Binary Translation, 2000

35 Steven P. Vanderwiel , David J. Lilja, Data prefetch mechanisms, ACM (CSUR), v.32 n.2, p.174-199, June 2000

↑ CITINGS 10

Jamison Collins , Suleyman Sair , Brad Calder , Dean M. Tullsen, Pointer cac prefetching, Proceedings of the 35th annual ACM/IEEE international sympos Microarchitecture, November 18-22, 2002, Istanbul, Turkey

Tatsushi Inagaki , Tamiya Onodera , Hideaki Komatsu , Toshio Nakatani, Str dynamically inspecting objects, ACM SIGPLAN Notices, v.38 n.5, May 2003

Ali-Reza Adl-Tabatabai , Richard L. Hudson , Mauricio J. Serrano , Sreeniva: Prefetch inection based on hardware monitoring and object metadata, ACM S v.39 n.6, May 2004

Matthias Hauswirth , Trishul M. Chilimbi, Low-overhead memory leak detect statistical profiling, Proceedings of the 11th international conference on Arch programming languages and operating systems, October 07-13, 2004, Boston

Robert Niewiadomski , José Nelson Amaral , Robert C. Holte, A performance techniques for improving data locality in refinement-based pathfinding, Journ Algorithmics (JEA), v.9 n.es, 2004

Erik Berg , Erik Hagersten, Fast data-locality profiling of native execution, A Performance Evaluation Review, v.33 n.1, June 2005

Xiangyu Zhang , Rajiv Gupta, Whole execution traces and their applications, on Architecture and Code Optimization (TACO), v.2 n.3, p.301-334, Septemb

Chen Ding , Yutao Zhong, Predicting whole-program locality through reuse d ACM SIGPLAN Notices, v.38 n.5, May 2003

Rodric M. Rabbah , Krishna V. Palem, Data remapping for design space optim memory systems, ACM Transactions on Embedded Computing Systems (TEC 218, May 2003

Matthew Arnold , Michael Hind , Barbara G. Ryder, Online feedback-directed Java, ACM SIGPLAN Notices, v.37 n.11, November 2002

↑ INDEX TERMS

Primary Classification:

D. Software

↳ D.3 PROGRAMMING LANGUAGES

↳ D.3.4 Processors

↳ Subjects: Code generation

Additional Classification:

D. Software

↳ D.3 PROGRAMMING LANGUAGES

↳ D.3.4 Processors

↳ Subjects: Optimization; Run-time environments

General Terms:

Measurement, Performance

Keywords:

data reference profiling, dynamic optimization, dynamic profiling, memory p
optimization, prefetching, temporal profiling

↑ Collaborative Colleagues:

Trishul M. Chilimbi: Thomas Ball

Bob Davidson

Stephen G. Eick

Matthias Hauswirth

Mark D. Hill

Martin Hirzel

James R. Larus

Martin Hirzel:

Trishul M. Chilimbi

Amer Diwan

Johannes Henkel

Matthew Hertz

Michael Hind

Antony L. Hosking

↑ **Peer to Peer - Readers of this Article have also read:**




- Constructing reality
Proceedings of the 11th annual international conference on Systems d
Douglas A. Powell , Norman R. Ball , Mansel W. Griffiths
- Data structures for quadtree approximation and compression
Communications of the ACM 28, 9
Hanan Samet
- A hierarchical single-key-lock access control using the Chinese remainder
Proceedings of the 1992 ACM/SIGAPP Symposium on Applied compi
Kim S. Lee , Huizhu Lu , D. D. Fisher
- An intelligent component database for behavioral synthesis
Proceedings of the 27th ACM/IEEE conference on Design automation
Gwo-Dong Chen , Daniel D. Gajski
- Putting innovation to work: adoption strategies for multimedia communic
Communications of the ACM 34, 12
Ellen Francik , Susan Ehrlich Rudman , Donna Cooper , Stephen Levine

↑ **This Article has also been published in:**

- ACM SIGPLAN Notices
Volume 37 , Issue 5 (May 2002)

The ACM Portal is published by the Association for Computing Machinery.
ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)



dynamic profile instrument binaries frequency 1990 - 2

Scholar Results 1 - 10 of about 1,740 for **dynamic profile instrument binari**

Dynamic frequency and voltage scaling for a multiple-clock-domain microprocessor - group of 7 »

[All articles](#) [Recent articles](#)

G Magklis, G Semeraro, DH Albonesi, SG Dropsho, S ... - Micro, IEEE, 2003 - ieeexplore.ieee.org

... a **binary** editing tool 8 to **instrument** subroutines and ... Both the online and

profile-based techniques that ... Using Multiple Clock Domains with **Dynamic Voltage** and ...

Cited by 7 - [Web Search](#) - [BL Direct](#)

Efficient path profiling - group of 31 »

T Ball, JR Larus - Proceedings of the 29th Annual International Symposium on ..., 1996 - doi.ieeeecs.org

... is proportional to the number of **dynamic** paths, rather than ... tree to select edges

to **instrument** and compute ... After collecting the run-time **profile**, derive the exe ...

Cited by 231 - [Web Search](#) - [BL Direct](#)

Exploiting hardware performance counters with flow and context sensitive profiling - group of 8 »

G Ammons, T Ball, JR Larus - ACM SIGPLAN Notices, 1997 - portal.acm.org

... is bounded, unlike a com- plete **dynamic** call tree ... To **instrument** a CFG for efficient

path profiling, it ... The algorithm only **profiles** paths that have no backedges ...

Cited by 123 - [Web Search](#) - [BL Direct](#)

Search strategies for Java bottleneck location by **dynamic** instrumentation - group of 4 »

DJ Brear, T Weise, T Wiffen, KC Yeung, SAM Bennett ... - Software, IEE

Proceedings-[see also Software Engineering, ..., 2003 - ieeexplore.ieee.org
... A more sophisti- cated **instrument** placement strategy ... of search
strategies for using
dynamic instrumentation to ... to explore automatic **profile**-driven
optimisation ...

Cited by 6 - Web Search - BL Direct

Calpa: a tool for automating selective **dynamic** compilation - group of 15 »
M Mock, C Chambers, SJ Eggers - Proceedings of the 33rd annual
ACM/IEEE international ..., 2000 - portal.acm.org
... benefit analysis, the instrumenter **instruments** an application ... of variables
and
instructions into static or **dynamic**. ... variance of value **profiles** across
different ...

Cited by 37 - Web Search - BL Direct

Bursty tracing: A framework for low-overhead temporal profiling - group of 7
»

M Hirzel, T Chilimbi - ... on Feedback-Directed and **Dynamic** Optimization
(FDDO-4), 2001 - cs.colorado.edu

... A **dynamic** optimizer that tries to complement a static ... We do not
instrument either

version of the code ... experiments, the quality of the **profile** actually
improved ...

Cited by 24 - View as HTML - Web Search

The concept of **dynamic** analysis - group of 11 »

T Ball - ESEC/SIGSOFT FSE, 1999 - Springer

... analysis applied to coverage **profiles** naturally computes ... as domination
and regions,

identifying “**dynamic** control flow ... analysis is a **binary** relation between ...

Cited by 67 - Web Search - BL Direct

Calpa: A tool for automating **dynamic** compilation - group of 6 »

M Mock, M Berryman, C Chambers, SJ Eggers - 2nd Workshop on Feedback-
Directed Optimization, 1999 - cs.washington.edu

... of a program is necessary in order to **profile** it, since they **instrument** the
executable ...

could also be used to drive an automatic **dynamic** compilation system ...

1

•

•

1

•

•

1

•

Goooooooooooooogle ▶

©2000 Google

[Workshop on]
16-18 Sept. 1995 Page(s): 481 - 490
Digital Object Identifier 10.1109/VLSISP
AbstractPlus | Full Text: [PDF\(420 KB\)](#)
[Rights and Permissions](#)

Indexed by
 Inspec

Home | Login | Logout


 IEEE Xplore[®]
RELEASE 2.1

Welcome United States Patent and Trademark Office

Search Results

[BROWSE SEARCH](#) [IEEE GUID](#)

Results for "(((call stack)<in>metadata)) <and> (pyr >= 1990 <and> py
Your search matched 15 of 1351636 documents.
A maximum of 100 results are displayed, 25 to a page, sorted by Relevance
Descending order.

» Search Options

[View Session History](#)
[New Search](#)

Modify Search

(((call stack)<in>metadata)) <and> (pyr >= 1990 <and> py

☐ Check to search only within this results set

» Key

IEEE JNL IEEE Journal or Magazine
IEE JNL IEE Journal or Magazine
IEEE CNF IEEE Conference Proceeding
IEE CNF IEE Conference Proceeding
IEEE STD IEEE Standard

Display Format: ☒ Citation ☐ Citation & Abstract

[view selected items](#)

[Select All](#) [Deselect All](#)

- ☐ 1. **Anomaly detection using call stack info**
Feng, H.H.; Kolesnikov, O.M.; Fogla, P.; Security and Privacy, 2003. Proceedings. 11-14 May 2003 Page(s):62 - 75
[AbstractPlus](#) | [Full Text: PDF\(361 KB\)](#) | [Rights and Permissions](#)
- ☐ 2. **Weighted median filters: a tutorial**
Lin Yin; Ruikang Yang; Gabbouj, M.; Ne Circuits and Systems II: Analog and Digital Transactions on [see also Circuits and Systems on] Volume 43, Issue 3, March 1996 Page(s) Digital Object Identifier 10.1109/82.4864
[AbstractPlus](#) | [Full Text: PDF\(5788 KB\)](#) | [Rights and Permissions](#)
- ☐ 3. **Comments on "A fast and efficient protocol for connected multicomputers"**
Lu Zhang; Computers, IEEE Transactions on Volume 52, Issue 2, Feb. 2003 Page(s):2 Digital Object Identifier 10.1109/TC.2003
[AbstractPlus](#) | [References](#) | [Full Text: PDF](#) | [Rights and Permissions](#)
- ☐ 4. **Using access control for secure information**
Banerjee, A.; Naumann, D.A.; Computer Security Foundations Workshop 30 June-2 July 2003 Page(s):155 - 169

[AbstractPlus](#) | [Full Text: PDF\(684 KB\)](#) | [Rights and Permissions](#)

- ☐ **5. Incremental stack-splitting mechanisms implementation of search-based AI syst**
 Villaverde, K.; Pontelli, E.; Guo, H.; Gupta
 Parallel Processing, International Conference
 3-7 Sept. 2001 Page(s):287 - 294
 Digital Object Identifier 10.1109/ICPP.2001.1000000
[AbstractPlus](#) | [Full Text: PDF\(712 KB\)](#) | [Rights and Permissions](#)
- ☐ **6. Bump-less interconnect for next genera**
 Suga, T.; Otsuka, K.;
 Electronic Components and Technology (ECTC)
 29 May-1 June 2001 Page(s):1003 - 1008
 Digital Object Identifier 10.1109/ECTC.2001.1000000
[AbstractPlus](#) | [Full Text: PDF\(1932 KB\)](#) | [Rights and Permissions](#)
- ☐ **7. On solving stack-based incremental satisfiability**
 Joonyoung Kim; Whitemore, J.; Sakallah
 Computer Design, 2000. Proceedings. 2000
 17-20 Sept. 2000 Page(s):379 - 382
 Digital Object Identifier 10.1109/ICCD.2000.1000000
[AbstractPlus](#) | [Full Text: PDF\(288 KB\)](#) | [Rights and Permissions](#)
- ☐ **8. Design and implementation of a program**
 Hu, M.; Vainio, O.; Gevorgian, D.;
 Multimedia and Expo, 2000. ICME 2000.
 on
 Volume 3, 30 July-2 Aug. 2000 Page(s):1
 Digital Object Identifier 10.1109/ICME.2000.1000000
[AbstractPlus](#) | [Full Text: PDF\(452 KB\)](#) | [Rights and Permissions](#)
- ☐ **9. Efficient large-scale process-oriented parallel**
 Perumalla, K.S.; Fujimoto, R.M.;
 Simulation Conference Proceedings, 1998
 Volume 1, 13-16 Dec. 1998 Page(s):459 - 464
 Digital Object Identifier 10.1109/WSC.1998.1000000
[AbstractPlus](#) | [Full Text: PDF\(672 KB\)](#) | [Rights and Permissions](#)
- ☐ **10. Multiprocessor architectures using multi**
 networks and distributed control
 Coudert, D.; Ferreira, A.; Munoz, X.;
 Parallel Processing Symposium, 1998. 19th
 First Merged International...and Symposium
 on Parallel Processing 1998.
 30 March-3 April 1998 Page(s):151 - 155
 Digital Object Identifier 10.1109/IPPS.1998.1000000
[AbstractPlus](#) | [Full Text: PDF\(96 KB\)](#) | [Rights and Permissions](#)
- ☐ **11. Committee pattern classifiers**
 Yu Hen Hu; Jong-Min Park; Knoblock, T.;
 Acoustics, Speech, and Signal Processing
 Conference Proceedings (ICASSP), 1998. The
 IEEE International Conference on

International Conference on
 Volume 4: 21-24 April 1997 Page(s):338
 Digital Object Identifier 10.1109/ICASSP
 AbstractPlus | Full Text: [PDF](#)(364 KB)
[Rights and Permissions](#)

- ☐ **12. Nonlinear committee pattern classification**
 Yu Hen Hu; Knoblock, T.; Jong-Ming Pa
 Neural Networks for Signal Processing [I
 IEEE Workshop
 24-26 Sept. 1997 Page(s):568 - 577
 Digital Object Identifier 10.1109/NNSP.1
 AbstractPlus | Full Text: [PDF](#)(516 KB)
[Rights and Permissions](#)

- ☐ **13. Threshold decomposition algorithm for operations. II. Erosion**
 Pu, C.C.;
 Image Processing and its Applications, 19
 on
 4-6 Jul 1995 Page(s):757 - 761
 AbstractPlus | Full Text: [PDF](#)(244 KB)

- ☐ **14. Java model checking**
 Park, D.Y.W.; Stern, U.; Skakkebaek, J.U.
 Automated Software Engineering, 2000. [I
 Fifteenth IEEE International Conference
 11-15 Sept. 2000 Page(s):253 - 256
 Digital Object Identifier 10.1109/ASE.20
 AbstractPlus | Full Text: [PDF](#)(320 KB)
[Rights and Permissions](#)

- ☐ **15. Understanding Java stack inspection**
 Wallach, D.S.; Felten, E.W.;
 Security and Privacy, 1998. Proceedings.
 3-6 May 1998 Page(s):52 - 63
 Digital Object Identifier 10.1109/SECPR
 AbstractPlus | Full Text: [PDF](#)(100 KB)
[Rights and Permissions](#)



USPTO

[Subscribe \(Full Service\)](#) [Register \(Limited Service\)](#)

Search: ☒ The ACM Digital Library ☐ The

+profil* +instrument* +frequently +dynamic*

[Feedback](#) [Report a problem](#) [Sign out](#)

Terms used **profil** **instrument** **frequently** **dynamic**

Sort results by [Save results to a Binder](#) [Try an Advanced Search](#)
[Search Tips](#) [Try this search in Google](#)

Display results ☐ Open results in a new window

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance

1 [Eliminating synchronization overhead in automatically parallelized programs](#)

[feedback](#)

Pedro C. Diniz, Martin C. Rinard

May 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17

Publisher: ACM Press

Full text available: [pdf](#) (244.57 KB) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

This article presents dynamic feedback, a technique that enables compilers to adapt dynamically to different execution environments. A compiler that uses dynamic feedback produces several different versions of the same source code; each version is optimized using a different optimization policy. The generated code alternately performs sampling and production phases. Each sampling phase measures the overhead of each version in the current environment. Each production phase uses the version with the lowest overhead.


Keywords: parallel computing, parallelizing compilers

2 [Profile-based optimizations: Dynamic trace selection using performance monitoring hardware sampling](#)


Howard Chen, Wei-Chung Hsu, Jiwei Lu, Pen-Chung Yew, Dong-Yuan Chao

March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization**


Publisher: IEEE Computer Society

Full text available:  [pdf\(1.88 MB\)](#) Additional Information: [full citation](#), [abstracts](#), [index terms](#)

Optimizing programs at run-time provides opportunities to apply aggressive optimizations to programs based on information that was not available at compile time. These programs can be adapted to better exploit architectural features, optimize code, use dynamic libraries, and simplify code based on run-time constants. Our framework provides a framework for collecting information required for performing optimizations. We sample the performance hardware registers available on


- 3 [Design and evaluation of dynamic optimizations for a Java just-in-time compiler](#)
 Toshio Suganuma, Toshiaki Yasue, Motohiro Kawahito, Hideaki Komatsu
July 2005 **ACM Transactions on Programming Languages and Systems**
Volume 27 Issue 4

Publisher: ACM Press

Full text available:  [pdf\(1.60 MB\)](#) Additional Information: [full citation](#), [abstracts](#), [index terms](#)

The high performance implementation of Java Virtual Machines (JVM) : (JIT) compilers is directed toward employing a dynamic compilation system of online runtime profile information. The trade-off between the compilation performance and the performance benefit is a crucial issue for such a system. This article describes the design and implementation of a dynamic optimization framework in a production Java compiler, together with two techniques for profile-directed optimization.

Keywords: JIT compiler, Recompilation, adaptive optimization, code space management, dynamic compilation, profile-directed method inlining

- 4 [Profiling Java applications using code hotswapping and dynamic call graph](#)
 Mikhail Dmitriev
January 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the international workshop on Software and performance Workshop**
Volume 29 Issue 1

Publisher: ACM Press


Full text available:  [pdf\(1.32 MB\)](#) Additional Information: [full citation](#), [abstracts](#)

Instrumentation-based profiling has many advantages and one serious disadvantage is usually high performance overhead. This overhead can be substantially reduced if only a small part of the target application (for example, one that has previously been identified as a performance bottleneck) is instrumented, while the rest of the application continues to run at full speed. The value of such a profiling technology would increase if the code could be instrumented and de-instrumented as needed.

5 Profile-directed optimization of event-based programs

- ◆ Mohan Rajagopalan, Saumya K. Debray, Matti A. Hiltunen, Richard D. Scott
May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN Conference on Programming language design and implementation '02**, Volume 37 Issue 5

Publisher: ACM Press

Full text available:  [pdf](#) (167.69 KB) Additional Information: [full citation](#), [abstracts](#), [citations](#), [index terms](#)


Events are used as a fundamental abstraction in programs ranging from graphical user interfaces (GUIs) to systems for building customized network protocols. As a flexible structuring and execution paradigm, events have the potential to reduce the drawback of extra execution overhead due to the indirection between method calls and events and those that handle them. This paper describes an approach to address this issue using static optimization techniques. This approach, which exploits the

Keywords: events, handlers, profiling

6 Continuous program optimization: A case study

- ◆ Thomas Kistler, Michael Franz
July 2003 **ACM Transactions on Programming Languages and Systems**, Volume 25 Issue 4

Publisher: ACM Press

Full text available:  [pdf](#) (877.67 KB) Additional Information: [full citation](#), [abstracts](#), [index terms](#), [reviews](#)

Much of the software in everyday operation is not making optimal use of the hardware on which it actually runs. Among the reasons for this discrepancy are hardware mismatches, modularization overheads introduced by software engineering practices, and the inability of systems to adapt to users' behaviors. A solution to the problem is to delay code generation until load time. This is the earliest point at which the

software can be fine-tuned to the actual capabilities of the ...


Keywords: Dynamic code generation, continuous program optimization
reoptimization

7 Practicing JUDO: Java under dynamic optimizations

◆ Michał Cierniak, Guei-Yuan Lueh, James M. Stichnoth

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLA
conference on Programming language design and implemen**
Volume 35 Issue 5

Publisher: ACM Press

Full text available:  pdf Additional Information: full citation, abstr
(190.06 KB) citings, index term


A high-performance implementation of a Java Virtual Machine (JVM) co
implementation of Just-In-Time (JIT) compilation, exception handling, s
mechanism, and garbage collection (GC). These components are tightly
achieve high performance. In this paper, we present some static and dyna
implemented in the JIT compilation and exception handling of the Micro
Research Lab Virtual Machine (MRL VM), ...

8 Dynamic hot data stream prefetching for general-purpose programs

◆ Trishul M. Chilimbi, Martin Hirzel

May 2002 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLA
Conference on Programming language design and implemer**
'02, Volume 37 Issue 5

Publisher: ACM Press

Full text available:  pdf Additional Information: full citation, abstr
(210.85 KB) citings, index term

Prefetching data ahead of use has the potential to tolerate the growing pr
performance gap by overlapping long latency memory accesses with use
While sophisticated prefetching techniques have been automated for lim
such as scientific codes that access dense arrays in loop nests, a similar l
has eluded general-purpose programs, especially pointer-chasing codes v
languages such as C and C++. We address this problem by describing ...

Keywords: data reference profiling, dynamic optimization, dynamic pro


performance optimization, prefetching, temporal profiling

9 Performance data collection using a hybrid approach

◆ Edu Metz, Raimondas Lencevicius, Teofilo F. Gonzalez

September 2005 **ACM SIGSOFT Software Engineering Notes**, **Proceedings of the European software engineering conference held jointly with the SIGSOFT international symposium on Foundations of software engineering ESEC/FSE-13**, Volume 30 Issue 5

Publisher: ACM Press

Full text available:  [pdf](#) (298.67 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)

Performance profiling consists of monitoring a software system during its execution and analyzing the obtained data. There are two ways to collect profiling data: through code instrumentation and statistical sampling. These two approaches have different advantages and drawbacks. This paper proposes a hybrid approach for performance data collection that combines the completeness of event tracing with the low overhead of statistical sampling. We propose to maximize the weighted amount of information collected.


Keywords: profiling, sampling, tracing

10 Enhancing software reliability with speculative threads

◆ Jeffrey Oplinger, Monica S. Lam

October 2002 **ACM SIGPLAN Notices**, **ACM SIGOPS Operating Systems Review**, **ACM SIGARCH Computer Architecture News**, **Proceedings of the international conference on Architectural support for programming languages and operating systems ASPLOS-X**, Volume 37 Issue 10, 5, 5

Publisher: ACM Press

Full text available:  [pdf](#) (1.47 MB) Additional Information: [full citation](#), [abstract](#), [citations](#)

This paper advocates the use of a monitor-and-recover programming paradigm to improve the reliability of software, and proposes an architectural design that allows hardware to cooperate in making this paradigm more efficient and easier to use. We propose that programmers write monitoring functions assuming simple execution semantics. Our architecture speeds up the computation by executing the monitoring functions in parallel with the program execution.


monitoring functions speculatively in parallel with the main computation

11 Low-overhead memory leak detection using adaptive statistical profiling

◆ Matthias Hauswirth, Trishul M. Chilimbi

October 2004 **ACM SIGPLAN Notices , ACM SIGOPS Operating Systems ACM SIGARCH Computer Architecture News , Proceedings of the international conference on Architectural support for programming languages and operating systems ASPLOS-XI, Volume 3**
11 , 5 , 5

Publisher: ACM Press

Full text available:  pdf (159.07 KB) Additional Information: [full citation](#), [abstracts](#), [citations](#), [index terms](#)

Sampling has been successfully used to identify performance optimization opportunities. We would like to apply similar techniques to check program correctness. Unfortunately, sampling provides poor coverage of infrequently executed code, where bugs often occur. We describe an adaptive profiling scheme that addresses this by sampling code segments at a rate inversely proportional to their execution frequency. Based on our ideas, we have implemented SWAT, a novel memory leak detector.


Keywords: low-overhead monitoring, memory leaks, runtime analysis

12 A portable sampling-based profiler for Java virtual machines

◆ John Whaley

June 2000 **Proceedings of the ACM 2000 conference on Java Grande**

Publisher: ACM Press


Full text available:  pdf(1.01 MB) Additional Information: [full citation](#), [references](#), [index terms](#)

13 Dynamic points-to sets: a comparison with static analyses and potential applications to program understanding and optimization

◆ Markus Mock, Manuvir Das, Craig Chambers, Susan J. Eggers

June 2001 **Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on programming language analysis for software tools and engineering**

Publisher: ACM Press


Full text available:  pdf (106.92 KB) Additional Information: [full citation](#), [abstracts](#), [index term](#)

In this paper, we compare the behavior of pointers in C programs, as app static pointer analysis algorithms, with the actual behavior of pointers w programs are run. In order to perform this comparison, we have impleme known pointer analysis algorithms, and we have built an instrumentation tracking pointer values during program execution. Our experiments show of programs from the Spec95 and Spec2000 benchma ...

Keywords: alias analysis, calpa, dynamic analysis, points-to analysis, pr instrumentation, program optimization, program understanding

- 14 Dynamic Adaptive compilation: An infrastructure for adaptive dynamic op
Derek Bruening, Timothy Garnett, Saman Amarasinghe
March 2003 **Proceedings of the international symposium on Code gene**
optimization: feedback-directed and runtime optimization

Publisher: IEEE Computer Society

Full text available:  pdf(1.16 MB) Additional Information: [full citation](#), [abstracts](#), [index term](#)


Dynamic optimization is emerging as a promising approach to overcome obstacles of traditional static compilation. But while there are a number of infrastructures for developing static optimizations, there are very few for dynamic optimizations. We present a framework for implementing dynam optimizations. We provide an interface for building external modules, or DynamoRIO dynamic code modification system. This interface abstracts

- 15 Targeted Path Profiling: Lower Overhead Path Profiling for Staged Dynam
Systems

Rahul Joshi, Michael D. Bond, Craig Zilles

March 2004 **Proceedings of the international symposium on Code gene**
optimization: feedback-directed and runtime optimization

Publisher: IEEE Computer Society

Full text available:  pdf (281.40 KB) Additional Information: [full citation](#), [abstracts](#)


In this paper, we present a technique for reducing the overhead of collect

in the context of a dynamic optimizer. The key idea to our approach, called Profiling (TPP), is to use an edge profile to simplify the collection of a portion of profile-guided profiling is a natural fit for dynamic optimizers, to optimize the code in a series of stages. TPP is an extension to the Ball-Lab Profile algorithm. Its increased efficiency ...

16 A region-based compilation technique for dynamic compilers

◆ Toshio Suganuma, Toshiaki Yasue, Toshio Nakatani
January 2006 **ACM Transactions on Programming Languages and Systems**
Volume 28 Issue 1

Publisher: ACM Press

Full text available:  [pdf](#) (977.63 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)


Method inlining and data flow analysis are two major optimization components of effective program transformations, but they often suffer from the existence of never executed code contained in the target method. One major problem is the assumption that the compilation unit is partitioned at method boundaries. This paper describes the design and implementation of a region-based compilation technique for a dynamic optimization framework, in which the compiled regions are selected based on the execution profile.

Keywords: JIT compiler, Region-based compilation, dynamic compilation, code replacement, partial inlining

17 The Accuracy of Initial Prediction in Two-Phase Dynamic Binary Translation

Youfeng Wu, Mauricio Breternitz, Justin Quek, Orna Etzion, Jesse Fang
March 2004 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization**

Publisher: IEEE Computer Society

Full text available:  [pdf](#) (234.04 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)

Dynamic binary translators use a two-phase approach to identify and optimize frequently executed code dynamically. In the first step (profiling phase), blocks of code are quickly translated to collect execution frequency information for the blocks. In the second phase (optimization phase), frequently executed blocks are grown and advanced optimizations are applied on them. This approach implicitly assumes that the initial profile of each block is representative of the block's execution profile.

18 Compilation and dynamic optimization: Dynamic parallelization and mapping executables on hierarchical platforms



Efe Yardimci, Michael Franz

May 2006 **Proceedings of the 3rd conference on Computing frontiers C**

Publisher: ACM Press

Full text available: [pdf](#) (241.09 KB) Additional Information: [full citation](#), [abstracts](#), [index terms](#)

As performance improvements are being increasingly sought via coarse-grained parallelism, established expectations of continued sequential performance are not being met. Current trends in computing point towards platforms seeking improvements through various degrees of parallelism, with coarse-grained features becoming commonplace in even entry-level systems. Yet the broad range of multiprocessor configurations that will be available that differ in the number of processors, the amount of shared memory, and the degree of interconnectivity, presents a challenge to the design of compilers and runtime systems. In this paper, we present a solution to the precision instrumentation problem for microcontroller code that is based upon our open, flexible simulator framework.

Keywords: continuous optimization, dynamic parallelization

19 Nonintrusive precision instrumentation of microcontroller software



Ben L. Titzer, Jens Palsberg

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Languages, compilers, and tools for embedded systems LCTES'05**, Volume 40 Issue 7

Publisher: ACM Press

Full text available: [pdf](#) (188.83 KB) Additional Information: [full citation](#), [abstracts](#), [index terms](#)


Debugging, testing, and profiling microcontroller programs are notoriously difficult due to the lack of supporting software such as an operating system, a narrow interface to the hardware chip, and delicately timed sequences of code present significant challenges. In this paper we present a solution to the precision instrumentation problem for microcontroller code that is based upon our open, flexible simulator framework.

Keywords: cycle-accurate, debugging, instruction-level simulation, instrumentation, monitoring, parallel simulation, profiling, sensor networks

20 Runtime specialization with optimistic heap analysis

◆ Ajeet Shankar, S. Subramanya Sastry, Rastislav Bodík, James E. Smith
October 2005 **ACM SIGPLAN Notices , Proceedings of the 20th annual
SIGPLAN conference on Object oriented programming,
languages, and applications OOPSLA '05**, Volume 40 Iss

Publisher: ACM Press

Full text available:  [pdf](#) (425.12 KB) Additional Information: [full citation](#), [abstr](#)
[index terms](#)

We describe a highly practical program specializer for Java programs. It is powerful, because it specializes optimistically, using (potentially transient) the heap; it is precise, because it specializes using data structures that are invariant; it is deployable, because it is hidden in a JIT compiler and does not require user annotations or offline preprocessing; it is simple, because it uses existing compiler ingredients; and it is fast, because ...




Keywords: dynamic optimization, partial evaluation, program analysis, ...

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#)

The ACM Portal is published by the Association for Computing Machinery.
ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media
Player](#)

[Home](#) | [Login](#) | [Logout](#)
IEEE Xplore[®]
RELEASE 2.1

Welcome United States Patent and Trademark Office

Search Results

[BROWSE SEARCH](#) [IEEE GUIDE](#)

Results for "(((frequently used instructions)<in>metadata)) <and> (pyr
<and> pyr <...
Your search matched 3 of 1351636 documents.
A maximum of 100 results are displayed, 25 to a page, sorted by **Relevance Descending** order.

» Search Options

[View Session History](#)
[New Search](#)

Modify Search

☐ Check to search only within this results set

» Key

IEEE JNL

 IEEE
Journal or Magazine

IEEE JNL

IEEE Journal or Magazine

IEEE CNF

 IEEE
Conference Proceeding

IEEE CNF

 IEEE
Conference Proceeding

IEEE STD

 IEEE
Standard

Display Format: ☒ Citation ☐ Citation & Abstract

[view selected items](#)
[Select All](#) [Deselect All](#)

- ☐ 1. **Code compression techniques using open**
Lin, K.; Chung, C.-P.; Computers and Digital Techniques, IEEE P.
Volume 149, Issue 1, Jan. 2002 Page(s):
Digital Object Identifier 10.1049/1b-cdt:200
AbstractPlus | Full Text: PDF(952 KB)
- ☐ 2. **Compiler-directed management of instr**
Chen, G.; Kayadif, I.; Zhang, W.; Kanden
Digital System Design, 2003. Proceedings
1-6 Sept. 2003 Page(s):459 - 462
Digital Object Identifier 10.1109/DSD.200
AbstractPlus | Full Text: PDF(278 KB)
Rights and Permissions
- ☐ 3. **68040 integer module**
Holden, K.; Eisele, R.; Kobe, M.; Raleigh
Computer Design: VLSI in Computers and
Proceedings., 1990 IEEE International Co
17-19 Sept. 1990 Page(s):183 - 186
Digital Object Identifier 10.1109/ICCD.19
AbstractPlus | Full Text: PDF(368 KB)
Rights and Permissions



[Home](#) | [Login](#) | [Logout](#)**IEEE Xplore**
RELEASE 2.1**Welcome United States Patent and
Trademark Office****Search Results****BROWSE SEARCH [IEEE](#)
[GUID](#)**

Results for "(((call stack)<in>metadata)) <and> (pyr >= 1990 <and> py
Your search matched 15 of 1351636 documents.
A maximum of 100 results are displayed, 25 to a page, sorted by **Relevance**
Descending order.

» Search Options

[View Session](#)
[History](#)
[New Search](#)

Modify Search(((call stack)<in>metadata)) <and> (pyr >= 1990 <ar☐ Check to search only within this results set**» Key**

IEEE IEEE
JNL Journal or
Magazine

IEE IEE Journal
JNL or Magazine

IEEE IEEE
CNF Conference
Proceeding

IEE IEE
CNF Conference
Proceeding

IEEE IEEE
STD Standard

Display ☒ Citation ☐ Citation &
Format: Abstract

[view selected items](#)[Select All](#) [Deselect All](#)

- ☐ 1. **Anomaly detection using call stack info**
Feng, H.H.; Kolesnikov, O.M.; Fogla, P.;
Security and Privacy, 2003. Proceedings.
11-14 May 2003 Page(s):62 - 75
AbstractPlus | Full Text: [PDF](#)(361 KB)
Rights and Permissions
- ☐ 2. **Weighted median filters: a tutorial**
Lin Yin; Ruikang Yang; Gabbouj, M.; Ne
Circuits and Systems II: Analog and Digit
Transactions on [see also Circuits and Sys
Transactions on]
Volume 43, Issue 3, March 1996 Page(s)
Digital Object Identifier 10.1109/82.4864
AbstractPlus | Full Text: [PDF](#)(5788 KB)
Rights and Permissions
- ☐ 3. **Comments on "A fast and efficient proc
connected multicomputers"**
Lu Zhang;
Computers, IEEE Transactions on
Volume 52, Issue 2, Feb. 2003 Page(s):2
Digital Object Identifier 10.1109/TC.2003
AbstractPlus | [References](#) | Full Text: [PDF](#)
Rights and Permissions
- ☐ 4. **Using access control for secure informa**
Banerjee, A.; Naumann, D.A.;
Computer Security Foundations Worksho
30 June-2 July 2003 Page(s):155 - 169

[AbstractPlus](#) | [Full Text: PDF\(684 KB\)](#) | [Rights and Permissions](#)

- ☐ **5. Incremental stack-splitting mechanisms implementation of search-based AI syst**
 Villaverde, K.; Pontelli, E.; Guo, H.; Gupta
 Parallel Processing, International Conference
 3-7 Sept. 2001 Page(s):287 - 294
 Digital Object Identifier 10.1109/ICPP.2001.1000000
[AbstractPlus](#) | [Full Text: PDF\(712 KB\)](#) | [Rights and Permissions](#)
- ☐ **6. Bump-less interconnect for next genera**
 Suga, T.; Otsuka, K.;
 Electronic Components and Technology Conference
 29 May-1 June 2001 Page(s):1003 - 1008
 Digital Object Identifier 10.1109/ECTC.2001.1000000
[AbstractPlus](#) | [Full Text: PDF\(1932 KB\)](#) | [Rights and Permissions](#)
- ☐ **7. On solving stack-based incremental satisf**
 Joonyoung Kim; Whittemore, J.; Sakallah
 Computer Design, 2000. Proceedings. 2000
 17-20 Sept. 2000 Page(s):379 - 382
 Digital Object Identifier 10.1109/ICCD.2000.1000000
[AbstractPlus](#) | [Full Text: PDF\(288 KB\)](#) | [Rights and Permissions](#)
- ☐ **8. Design and implementation of a progra**
 Hu, M.; Vainio, O.; Geyorkian, D.;
 Multimedia and Expo, 2000. ICME 2000.
 on
 Volume 3, 30 July-2 Aug. 2000 Page(s):1
 Digital Object Identifier 10.1109/ICME.2000.1000000
[AbstractPlus](#) | [Full Text: PDF\(452 KB\)](#) | [Rights and Permissions](#)
- ☐ **9. Efficient large-scale process-oriented pa**
 Perumalla, K.S.; Fujimoto, R.M.;
 Simulation Conference Proceedings, 1998
 Volume 1, 13-16 Dec. 1998 Page(s):459 - 464
 Digital Object Identifier 10.1109/WSC.1998.1000000
[AbstractPlus](#) | [Full Text: PDF\(672 KB\)](#) | [Rights and Permissions](#)
- ☐ **10. Multiprocessor architectures using mu**
 networks and distributed control
 Coudert, D.; Ferreira, A.; Munoz, X.;
 Parallel Processing Symposium, 1998. 19
 First Merged International...and Symposium
 Processing, 1998.
 30 March-3 April 1998 Page(s):151 - 155
 Digital Object Identifier 10.1109/IPPS.1998.1000000
[AbstractPlus](#) | [Full Text: PDF\(96 KB\)](#) | [Rights and Permissions](#)
- ☐ **11. Committee pattern classifiers**
 Yu Hen Hu; Jong-Min Park; Knoblock, T.
 Acoustics, Speech, and Signal Processing

International Conference on
Volume 4, 21-24 April 1997 Page(s):338
Digital Object Identifier 10.1109/ICASSP
AbstractPlus | Full Text: [PDF\(364 KB\)](#)
Rights and Permissions

- ☐ **12. Nonlinear committee pattern classification**
 Yu Hen Hu; Knoblock, T.; Jong-Ming Pa
 Neural Networks for Signal Processing [I
 IEEE Workshop
 24-26 Sept. 1997 Page(s):568 - 577
 Digital Object Identifier 10.1109/NNSP.1
 AbstractPlus | Full Text: [PDF\(516 KB\)](#)
 Rights and Permissions

- ☐ **13. Threshold decomposition algorithm for operations. II. Erosion**
 Pu, C.C.;
 Image Processing and its Applications, 19
 on
 4-6 Jul 1995 Page(s):757 - 761
 AbstractPlus | Full Text: [PDF\(244 KB\)](#)

- ☐ **14. Java model checking**
 Park, D.Y.W.; Stern, U.; Skakkebaek, J.U.
 Automated Software Engineering, 2000. [I
 Fifteenth IEEE International Conference
 11-15 Sept. 2000 Page(s):253 - 256
 Digital Object Identifier 10.1109/ASE.20
 AbstractPlus | Full Text: [PDF\(320 KB\)](#)
 Rights and Permissions

- ☐ **15. Understanding Java stack inspection**
 Wallach, D.S.; Felten, E.W.;
 Security and Privacy, 1998. Proceedings.
 3-6 May 1998 Page(s):52 - 63
 Digital Object Identifier 10.1109/SECPR
 AbstractPlus | Full Text: [PDF\(100 KB\)](#)
 Rights and Permissions